Fast Adaptive Bilateral Filtering of Color Images

Ruturaj G. Gavaskar and Kunal N. Chaudhury

Department of Electrical Engineering,
Indian Institute of Science

IEEE International Conference on Image Processing, Taipei (2019)

Nonlinear edge-preserving smoothing[1]:

$$\boldsymbol{g}(\boldsymbol{i}) = \eta(\boldsymbol{i})^{-1} \sum_{\boldsymbol{j}\in\Omega} \omega(\boldsymbol{j})\phi\big(\boldsymbol{f}(\boldsymbol{i}-\boldsymbol{j}) - \boldsymbol{f}(\boldsymbol{i})\big)\boldsymbol{f}(\boldsymbol{i}-\boldsymbol{j}),$$

$$\eta(\boldsymbol{i}) = \sum_{\boldsymbol{j}\in\Omega} \omega(\boldsymbol{j})\phi\big(\boldsymbol{f}(\boldsymbol{i}-\boldsymbol{j}) - \boldsymbol{f}(\boldsymbol{i})\big),$$

where

- $\boldsymbol{f}$ and $\boldsymbol{g}$ are the input and output RGB images.
- $\boldsymbol{f}(\boldsymbol{i})$ and $\boldsymbol{g}(\boldsymbol{i})$ are vectors.
- ω and ϕ = Gaussian kernels with variance ρ^2 and σ^2.
- Ω = Neighborhood for averaging.

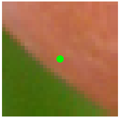
[1] Tomasi and Manduchi, 1998

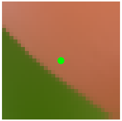
Role of σ

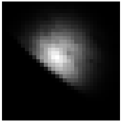

Input.

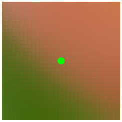

Output, $\sigma = 30$.

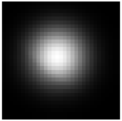

Output, $\sigma = 200$.






Weights




Weights

Adaptation of σ

- σ (width of range kernel) controls the extent of blurring.
- A fixed σ either over or under smooths.
- Useful for controlling the blur in different regions, e.g., more blur to remove coarse textures in images.
- σ is allowed to change at each pixel (a rule is required).
- Proposed for a couple of applications (for grayscale images):
  - Image sharpening[2].
  - JPEG deblocking[3].

[2] Zhang and Allebach, 2008.
[3] Zhang and Gunturk, 2009.

- Make the width of the range kernel a function of $\boldsymbol{i}$.
- Moreover, allow center[4] to be different from $\boldsymbol{f}(\boldsymbol{i})$.

$$\boldsymbol{g}(\boldsymbol{i}) = \eta(\boldsymbol{i})^{-1} \sum_{\boldsymbol{j}\in\Omega} \omega(\boldsymbol{j})\ \phi_{\boldsymbol{i}}\big(\boldsymbol{f}(\boldsymbol{i}-\boldsymbol{j}) - \boldsymbol{\theta}(\boldsymbol{i})\big)\boldsymbol{f}(\boldsymbol{i}-\boldsymbol{j}),$$

$$\eta(\boldsymbol{i}) = \sum_{\boldsymbol{j}\in\Omega} \omega(\boldsymbol{j})\ \phi_{\boldsymbol{i}}\big(\boldsymbol{f}(\boldsymbol{i}-\boldsymbol{j}) - \boldsymbol{\theta}(\boldsymbol{i})\big)\boldsymbol{f}(\boldsymbol{i}-\boldsymbol{j}).$$

- However, a fixed spatial kernel is used.

[4] Zhang and Allebach, 2008.

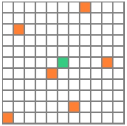
Computation cost

- $O(\rho^2)$ computations per pixel.
- Higher ρ (window size) is used for higher-resolution images.
- e.g. 60 seconds for a 2 megapixel image on a CPU.
- Real-time implementation is challenging.
- Fast approximation: Approximate the original formula and hope to speed it up, without appreciable loss of visual information.

- Several fast algorithms for classical bilateral filtering (gray/color).
- Complexity does not scale with filter width ($O(1)$ implementation).
- Almost all fundamentally require the range kernel to be fixed.
- Filtering reduced to fast convolutions by approximating the range kernel.
- Rules out extension to ABF (range kernel is changing).

- Novel $O(1)$ algorithm for fast ABF of color images.
- Builds on a recently proposed algorithm for gray images[5].
- Trivial channel-by-channel extension to color images (3X cost).
- Filtering in RGB space?
- As explained later, this poses technical challenges.
- Core idea: Express filtering using local (weighted) histograms[6].

[5] Gavaskar and Chaudhury, 2019.
[6] Mozerov and van de Weijer, 2015.



- Local histogram at pixel $\boldsymbol{i}$:

$$h_{\boldsymbol{i}}(\boldsymbol{t}) = \sum_{\boldsymbol{j}\in\Omega} \delta\big(\boldsymbol{f}(\boldsymbol{i}-\boldsymbol{j}) - \boldsymbol{t}\big), \quad \boldsymbol{t} \in \{0,\ldots,255\}^3.$$

- $\boldsymbol{t} = (t_r, t_g, t_b)$ and $\delta(\boldsymbol{t}) = \delta(t_r)\ \delta(t_g)\ \delta(t_b)$.

- Local weighted histogram at pixel $\boldsymbol{i}$:

$$h_{\boldsymbol{i}}(\boldsymbol{t}) = \sum_{\boldsymbol{j}\in\Omega} \omega(\boldsymbol{j})\ \delta\big(\boldsymbol{f}(\boldsymbol{i}-\boldsymbol{j}) - \boldsymbol{t}\big), \quad \boldsymbol{t} \in \{0,\ldots,255\}^3.$$

- Interpretation: Spatially-weighted frequency of RGB value $\boldsymbol{t}$.

ABF in terms of local weighted histograms:

$$\boldsymbol{g}(\boldsymbol{i}) = \eta(\boldsymbol{i})^{-1} \sum_{\boldsymbol{t}} \boldsymbol{t} h_{\boldsymbol{i}}(\boldsymbol{t}) \phi_{\boldsymbol{i}}\big(\boldsymbol{t} - \boldsymbol{\theta}(\boldsymbol{i})\big),$$

and

$$\eta(\boldsymbol{i}) = \sum_{\boldsymbol{t}} h_{\boldsymbol{i}}(\boldsymbol{t}) \phi_{\boldsymbol{i}}\big(\boldsymbol{t} - \boldsymbol{\theta}(\boldsymbol{i})\big),$$

where sum is over RGB values in the neighborhood of $\boldsymbol{i}$.

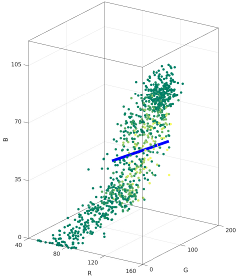
- ABF for grayscale images can be similarly reformulated.
- In grayscale, $h_{\mathbf{i}}(t)$ is a function of a scalar variable.
- For fast algorithm, $h_{\mathbf{i}}(t)$ is approximated using polynomials[7].
- This gave closed-form Gaussian integrals.
- Histogram approximation using fast convolutions (moment matching).
- For color images, $h_{\mathbf{i}}(\boldsymbol{t})$ is a function of a vector variable.
- Polynomial approximation is bad due to sparse data.

[7] Gavaskar and Chaudhury, 2019.

- Motivated by the approach in Mozerov and van de Weijer[8]:
  - $h_{\boldsymbol{i}}(\boldsymbol{t})$ is constant over an interval $[\boldsymbol{a_i}, \boldsymbol{b_i}]$ (in $\mathbb{R}^3$).
  - $h_{\boldsymbol{i}}(\boldsymbol{t})$ is zero elsewhere.
  - Summations are replaced by line integrals:

$$\hat{\boldsymbol{g}}(\boldsymbol{i}) = \hat{\eta}(\boldsymbol{i})^{-1} \int_{[\boldsymbol{a_i},\boldsymbol{b_i}]} \boldsymbol{t}\ \phi_{\boldsymbol{i}}\big(\boldsymbol{t} - \boldsymbol{\theta}(\boldsymbol{i})\big)\, d\boldsymbol{t},$$

$$\hat{\eta}(\boldsymbol{i}) = \int_{[\boldsymbol{a_i},\boldsymbol{b_i}]} \phi_{\boldsymbol{i}}\big(\boldsymbol{t} - \boldsymbol{\theta}(\boldsymbol{i})\big)\, d\boldsymbol{t}.$$



- The integrals, and hence the filter, have a closed-form expression.
- By clever choice of the interval, the computation becomes $O(1)$.

[8] Mozerov and van de Weijer, 2015.

Novelty of our proposal

- In Mozerov and van de Weijer, the interval was chosen to be
  - passing through $\boldsymbol{f}(\boldsymbol{i})$.
  - having direction $\bar{\boldsymbol{f}}(\boldsymbol{i}) - \boldsymbol{f}(\boldsymbol{i})$, where $\bar{\boldsymbol{f}}(\boldsymbol{i}) =$ mean value.
- This makes the algorithm $O(1)$, but is an ad-hoc choice.
- We choose the interval such that it captures linear trend of data.
- To do this, we use the covariance of the local weighted histogram.
- Our proposed algorithm is also $O(1)$.

- Covariance matrix:

$$\mathrm{C}_{\boldsymbol{i}} = \sum_{\boldsymbol{j}\in\Omega} \omega(\boldsymbol{j})\big(\boldsymbol{f}(\boldsymbol{i}-\boldsymbol{j}) - \bar{\boldsymbol{f}}(\boldsymbol{i})\big)\big(\boldsymbol{f}(\boldsymbol{i}-\boldsymbol{j}) - \bar{\boldsymbol{f}}(\boldsymbol{i})\big)^{\top}.$$

- Direction of $[\boldsymbol{a}_{\boldsymbol{i}}, \boldsymbol{b}_{\boldsymbol{i}}]$ = Largest eigenvector of the covariance matrix.

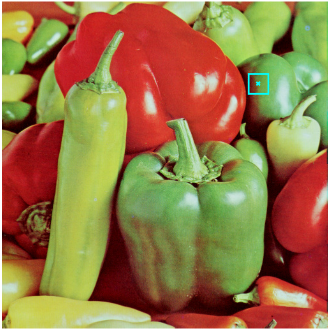
- This should give "best" linear approximation of the set of data points.

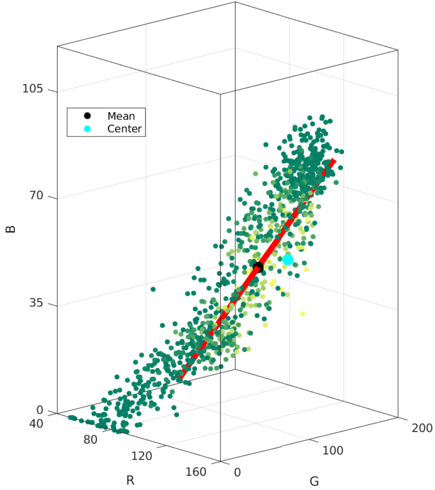
- Proposal:

$$[\boldsymbol{a}_{\boldsymbol{i}}, \boldsymbol{b}_{\boldsymbol{i}}] = \Big[\bar{\boldsymbol{f}}(\boldsymbol{i}) - c\sqrt{\lambda_{\boldsymbol{i}}}\ \boldsymbol{q}_{\boldsymbol{i}}, \bar{\boldsymbol{f}}(\boldsymbol{i}) + c\sqrt{\lambda_{\boldsymbol{i}}}\ \boldsymbol{q}_{\boldsymbol{i}}\Big];$$

$$(\lambda_{\boldsymbol{i}}, \boldsymbol{q}_{\boldsymbol{i}}) = \text{Top eigenpair of } \mathrm{C}_{\boldsymbol{i}},$$

$$c = \text{Positive constant, decides length of the interval.}$$





Mean
Center
B
105
70
35
0
R
40
80
120
160
G
0
100
200

Fast computation of interval endpoints

- Recall: $\boldsymbol{a_i} = \bar{\boldsymbol{f}}(\boldsymbol{i}) - c\sqrt{\lambda_{\boldsymbol{i}}}\ \boldsymbol{q_i}$, $\boldsymbol{b_i} = \bar{\boldsymbol{f}}(\boldsymbol{i}) + c\sqrt{\lambda_{\boldsymbol{i}}}\ \boldsymbol{q_i}$.
- We must find a fast method to compute the end points.
- $O(1)$ Gaussian convolutions come to our rescue.
- $\bar{\boldsymbol{f}}(\boldsymbol{i}) = \omega * \boldsymbol{f}(\boldsymbol{i}) \rightarrow 3$ Gaussian convolutions.
- (p, q)th entry of $\mathrm{C}_{\boldsymbol{i}} = \omega * (f_p f_q)(i) - \big(\omega * f_p(i)\big)\ \big(\omega * f_q(i)\big)$.
- 6 additional Gaussian convolutions to compute $\mathrm{C}_{\boldsymbol{i}}$'s.

Fast computation of interval endpoints

- $(\lambda_{\boldsymbol{i}}, \boldsymbol{q_i})$ computed using power iterations method.
- Power iterations:
  - Initialize $\boldsymbol{q_i}$ as unit vector along $\bar{\boldsymbol{f}}(\boldsymbol{i}) - \boldsymbol{f}(\boldsymbol{i})$.[9]
  - Iterate: $\boldsymbol{q_i} \leftarrow \mathsf{C}_{\boldsymbol{i}} \boldsymbol{q_i} / \|\boldsymbol{q_i}\|$.
- In practice, just one iteration is enough.
- $\lambda_{\boldsymbol{i}} = \boldsymbol{q_i}^\top \mathsf{C}_{\boldsymbol{i}} \boldsymbol{q_i}$.
- Overall, computation of $\boldsymbol{a_i}, \boldsymbol{b_i}$ requires $O(1)$ operations.

[9] Direction used in Mozerov and van de Weijer, 2015.

- Recall:

$$\hat{\boldsymbol{g}}(\boldsymbol{i}) = \hat{\eta}(\boldsymbol{i})^{-1} \int_{[\boldsymbol{a_i}, \boldsymbol{b_i}]} \boldsymbol{t}\, \phi_{\boldsymbol{i}}\big(\boldsymbol{t} - \boldsymbol{\theta}(\boldsymbol{i})\big)\, d\boldsymbol{t},$$

$$\hat{\eta}(\boldsymbol{i}) = \int_{[\boldsymbol{a_i}, \boldsymbol{b_i}]} \phi_{\boldsymbol{i}}\big(\boldsymbol{t} - \boldsymbol{\theta}(\boldsymbol{i})\big)\, d\boldsymbol{t}.$$

- The integrals have closed-form expressions in terms of $\boldsymbol{a_i}, \boldsymbol{b_i}$.
- This was made possible due to the nature of the approximation.
- As computation of $\boldsymbol{a_i}, \boldsymbol{b_i}$ is $O(1)$, computation of $\hat{\boldsymbol{g}}(\boldsymbol{i})$ becomes $O(1)$.

- Closed-form expression (mean + first-order correction):

$$\hat{\boldsymbol{g}}(\boldsymbol{i}) = \bar{\boldsymbol{f}}(\boldsymbol{i}) + \left(2\left(\beta - \alpha e_1 e_2^{-1}\right) - 1\right) c\sqrt{\lambda_{\boldsymbol{i}}}\,\boldsymbol{q}_{\boldsymbol{i}},$$

where

$$\begin{aligned}
\alpha &= \sigma(\boldsymbol{i})/c\sqrt{2\pi\lambda_{\boldsymbol{i}}}, \\
\beta &= \frac{1}{2c\sqrt{\lambda_{\boldsymbol{i}}}}\boldsymbol{q}_{\boldsymbol{i}}^{\top}\left(\boldsymbol{\theta}(\boldsymbol{i}) - \bar{\boldsymbol{f}}(\boldsymbol{i}) + c\sqrt{\lambda_{\boldsymbol{i}}}\mathrm{q}_{\boldsymbol{i}}\right), \\
e_1 &= \exp\left(-\frac{(1-\beta)^2}{\pi\alpha^2}\right) - \exp\left(-\frac{\beta^2}{\pi\alpha^2}\right), \\
e_2 &= \operatorname{erf}\left(\frac{1-\beta}{\sqrt{\pi}\alpha}\right) - \operatorname{erf}\left(-\frac{\beta}{\sqrt{\pi}\alpha}\right).
\end{aligned}$$

- Main point: All computations are $O(1)$.

1. Compute $\omega * (f_p f_q)$, $\omega * f_p$ for $p, q = 1, 2, 3$ using $O(1)$ convolutions.

2. For each pixel $\boldsymbol{i}$,

   2.1 Populate $\mathrm{C}_{\boldsymbol{i}}$ using the above convolved quantities.

   2.2 Estimate dominant eigenpair $(\lambda_{\boldsymbol{i}}, \boldsymbol{q}_{\boldsymbol{i}})$ by power iterations method.

   2.3 Compute α, β, e_1, e_2 in the previous slide.

   2.4 Compute $\hat{\boldsymbol{g}}(\boldsymbol{i})$ using the formula in the previous slide.

Dominant cost = 9 Gaussian convolutions.

Brief overview:

- Objective: Enhance details, but not to the same extent everywhere.
- More enhancement in regions which are more visually salient.
- Can be accomplished using the ABF[10].
- $\sigma(\boldsymbol{i})$ is decided using a saliency map.
- $\boldsymbol{\theta}(\boldsymbol{i}) = \boldsymbol{f}(\boldsymbol{i})$.
- We use our proposed algorithm for color filtering.

[10] Ghosh et al., 2019.

Input (640×960).

Enhanced, $\rho = 5$.

Saliency map.

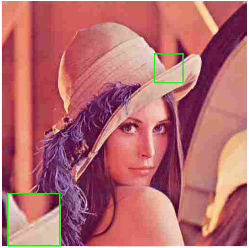
σ map.

Timings: Brute-force $=$ 27 sec., Proposed $=$ 1.4 sec.

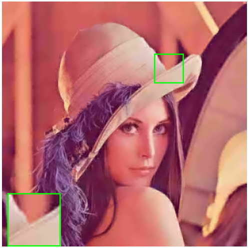
Brief overview:

- Objective: Smooth out blocking artifacts in JPEG-compressed images.
- For grayscale images, can be accomplished using ABF[11][12].
- We extend the same idea to color images.
- $\sigma(\boldsymbol{i})$ is decided using a technique proposed previously [11].
- $\boldsymbol{\theta}(\boldsymbol{i}) = \boldsymbol{f}(\boldsymbol{i})$.
- We use our proposed algorithm for filtering.

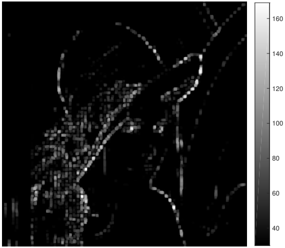
[11] Zhang and Gunturk, 2009.
[12] Gavaskar and Chaudhury, 2019.



Input (512×512).



Deblocked, $\rho = 4$.



σ map.

Original.

Timings: Brute-force $= 8.4$ sec., Proposed $= 0.6$ sec.

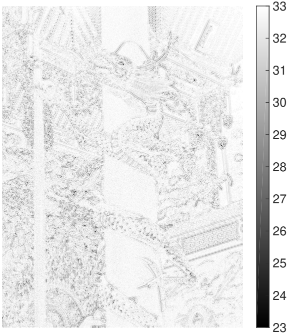
Brief overview:

- Objective: Sharpen a blurred image containing fine noise grains.
- For grayscale images, can be accomplished using ABF[13].
- We extend the idea to color images.
- Both $\sigma(\boldsymbol{i})$ and $\boldsymbol{\theta}(\boldsymbol{i})$ are decided using previously proposed techniques.
- We use our proposed algorithm for filtering.

[13] Zhang and Allebach, 2008.

Input (1600 $\times$ 1200).

Sharpened, $\rho = 4$.



σ map.

Timings: Brute-force = 62 sec., Proposed = 4.4 sec.

Conclusion

- Proposed $O(1)$ algorithm for adaptive bilateral filtering of color images.
- First such algorithm to the best of our knowledge.
- Core idea: Approximate local histogram as uniform along direction of maximum variance.
- Achieves about $15\times$ speedup with reasonable accuracy.
- Useful for detail enhancement, sharpening, and deblocking.
- Better accuracy and extension to non-Gaussian kernels?

Research supported by EMR grant SERB/F/6047/2016-2017 from DST-SERB, Government of India.

References I

- C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," Proc. IEEE International Conference on Computer Vision, pp. 839–846, 1998.

- B. Zhang and J. P. Allebach, "Adaptive bilateral filter for sharpness enhancement and noise removal," IEEE Transactions on Image Processing, vol. 17, no. 5, pp. 664–678, 2008.

- K. Sugimoto and S.-I. Kamata, "Compressive bilateral filtering," IEEE Transactions on Image Processing, vol. 24, no. 11, pp. 3357–3369, 2015.

- M. G. Mozerov and J. van de Weijer, "Global color sparseness and a local statistics prior for fast bilateral filtering," IEEE Transactions on Image Processing, vol. 24, no. 12, pp. 5842–5853, 2015.

- R. Deriche, "Recursively implementing the Gaussian and its derivatives," Research Report RR-1893, INRIA, 1993.

- I. T. Young and L. J. van Vliet, "Recursive implementation of the Gaussian filter," Signal Processing, vol. 44, pp. 139–151, 1995.

- S. Ghosh, R. G. Gavaskar, and K. N. Chaudhury, "Saliency guided image detail enhancement," Proc. National Conference on Communications, pp. 1–6, 2019.

- M. Zhang and B. K. Gunturk, "Compression artifact reduction with adaptive bilateral filtering," Proc. SPIE Visual Communications and Image Processing, vol. 7257, 2009.

- R. G. Gavaskar and K. N. Chaudhury, "Fast adaptive bilateral filtering," IEEE Transactions on Image Processing, vol. 28, no. 2, pp. 779–790, 2019.

Thanks for listening!